
SERIALIZING VALUE OBJECTS IN RAILS

ARA HACOPIAN

@ahacop
SmartLogic
© 2015

FAT MODELS

“[An] argument against Active Record is the fact that it couples the object design to the database design. This makes it more difficult to refactor either design...”

“[An] argument against Active Record is the fact that it couples the object design to the database design. This makes it more difficult to refactor either design...”

-- Martin Fowler, "Patterns of Enterprise Application Architecture", inventor of the Active Record pattern

SINGLE RESPONSIBILITY PRINCIPLE

THERE SHOULD NEVER BE MORE THAN ONE REASON FOR A CLASS TO CHANGE

ALL THE METHODS THAT HAVE TO DO WITH A PERSON GO INTO THE PERSON MODEL

```ruby
class Person < ActiveRecord::Base
  def initials
    first_name.chr + middle_name.chr + last_name.chr
  end

  def full_name
    "#{title} #{first_name} #{middle_name} #{last_name}, #{suffix}"
  end

  def weight_in_pounds
    weight
  end

  def weight_in_kilograms
    (weight * 0.453592).round
  end

  def weight_in_stone
    (weight * 0.07142).round
  end
end
```

COHESION

"WHERE SHOULD THIS METHOD GO?"

```ruby
class Person < ActiveRecord::Base
  def initials
    first_name.chr + middle_name.chr + last_name.chr
  end

  def full_name
    "#{title} #{first_name} #{middle_name} #{last_name}, #{suffix}"
  end

  def weight_in_pounds
    weight
  end

  def weight_in_kilograms
    (weight * 0.453592).round
  end

  def weight_in_stone
    (weight * 0.07142).round
  end
end
```

HAS_ONE?

```
class Person < ActiveRecord::Base
  has_one :weight
end

class Weight < ActiveRecord::Base
  belongs_to :person
end
```

VALUE OBJECTS

```ruby
class Weight
  include Comparable

  def <=>(other)
    self.class == other.class && pounds <=> other.pounds
  end

  def hash
    pounds.hash
  end
end
```

SERIALIZE

```ruby
class Person < ActiveRecord::Base
  def weight_in_pounds
    weight
  end

  def weight_in_kilograms
    (weight * 0.453592).round
  end

  def weight_in_stone
    (weight * 0.07142).round
  end
end
```

```ruby
class Person < ActiveRecord::Base
  serialize :weight, Weight
end
```

```ruby
class Weight
  class << self
    def dump(weight)
      weight.pounds
    end

    def load(pounds)
      new(pounds)
    end
  end

  def initialize(pounds)
    @pounds = pounds
  end
```

```ruby
  attr_reader :pounds

  def kilograms
    (pounds * 0.453592).round
  end

  def stone
    (pounds * 0.07142).round
  end
end
```

```ruby
class Weight
  include Comparable

  attr_reader :pounds

  class << self
    def dump(weight)
      weight.pounds
    end

    def load(pounds)
      new(pounds)
    end
  end

  def initialize(pounds)
    @pounds = pounds
  end

  def kilograms
    (pounds * 0.453592).round
  end

  def stone
    (pounds * 0.07142).round
  end

  def <=>(other)
    self.class == other.class && pounds <=> other.pounds
  end

  def hash
    pounds.hash
  end
end
```

USING THE VALUE OBJECT WITH ACTIVERECORD

```
weight = Weight.new(150)
person = Person.create(weight: weight)
```

VIRTUAL ATTRIBUTES

```ruby
class Weight
  include Comparable

  attr_reader :pounds

  def initialize(pounds)
    @pounds = pounds
  end

  def kilograms
    (pounds * 0.453592).round
  end

  def stone
    (pounds * 0.0714286).round
  end

  def <=>(other)
    self.class == other.class && pounds <=> other.pounds
  end

  def hash
    pounds.hash
  end
end
```

```
Person < ActiveRecord::Base
  def weight
    @weight ||= Weight.new(weight_value)
  end

  def weight=(other_weight)
    self.weight_value = other_weight.pounds

    @weight = other_weight
  end
end
```

```ruby
class Person < ActiveRecord::Base
  def name
    @name ||= Name.new(title, first_name, middle_name, last_name,
suffix)
  end

  def name=(other_name)
    self.title = other_name.title
    self.first_name = other_name.first
    self.middle_name = other_name.middle
    self.last_name = other_name.last
    self.suffix = other_name.suffix

    @name = other_name
  end
end
```

COMPOSED_OF

```ruby
class Person < ActiveRecord::Base
  composed_of :name,
    allow_nil: true,
    mapping: [
      %w(title title),
      %w(first_name first),
      %w(middle_name middle),
      %w(last_name last),
      %w(suffix suffix)
    ]

  composed_of :weight,
    allow_nil: true,
    mapping: %w(weight_value pounds)
end
```

```ruby
class Name
  attr_reader :title, :first, :middle, :last, :suffix

  def initialize(title, first, middle, last, suffix)
    @title, @first, @middle, @last, @suffix = title, first, middle,
last, suffix
  end

  def initials
    first.chr + middle.chr + last.chr
  end

  def full_name
    "#{title} #{first} #{middle} #{last}, #{suffix}"
  end
```

```ruby
  def ==(other)
    self.class == other.class &&
      title == other.title &&
      first == other.first &&
      middle == other.middle &&
      last == other.last &&
      suffix == other.suffix
  end

  def hash
    @hash ||= [title, first, middle, last, suffix].hash
  end
end
```

QUESTIONS?